In re Application of MOORE et al.
Application No. 09/755,770

Amendments to the Claims

1. (Currently Amended) A method of preventing blocking of an application communicating with another device utilizing a connection, wherein the application comprises an application-client configured to communicate asynchronously with a server and ~~has~~ a user interface for accepting a user input from and presenting an output to a user, the user input including at least one command requiring communication with ~~a~~ the server, wherein the user input is handled by the application, the method comprising the steps of:

forwarding, by the application[[-client]], a user request to the user interface of the application to ~~an~~ the application-client of the application, the user request comprising the at least one command;

issuing, by the application-client of the application, a confirmation message to complete a request-acknowledgment loop between the user interface of the application and the application-client of the application prior to executing the request thereby freeing the user interface of the application to process subsequent user input prior to completion of the request;

storing user input from the user interface of the application for subsequent handling by ~~an~~ the application-client of the application; and

communicating, by the application-client of the application, with the server to handle the user input received from the user interface of the application.

2. (Original) The method of claim 1 having the additional steps of estimating an error rate for successfully transmitting data of interest over the connection; and selecting a frame size based upon the error rate.

3. (Original) The method of claim 1 having the additional steps of estimating a bandwidth-delay, due to link and network congestion, for successfully transmitting data of interest over the connection; and selecting a frame size based upon the bandwidth-delay.

4. (Original) The method of claim 1 further including the step of using a default frame size as the frame size if an error rate is not available.

5. (Original) The method of claim 1 further including the step of organizing data to be transmitted in a transaction into functional segments; and defining a state of the application-client and a state of the server communicating over the connection by functional segments.

6. (Original) The method of claim 5 further including the step of determining the state of the application-client by referencing locally stored functional segments.

7. (Original) The method of claim 6 including a description of a step of providing the state of the application-client to the server transmitted to the application-client in the transaction.

8. (Original) The method of claim 1 further including the step of determining the state of the server by identifying functional segments already available locally at the server.

9. (Original) The method of claim 8 further including the step of providing the state of the server to the application-client to determine a set of remaining functional segments to be transmitted to the server in the transaction.

10. (Original) The method of claim 5 further including the step of updating the state of the application-client and the state of the server during a transaction over the connection.

11. (Original) The method of claim 6 further including the step of updating the state of the application-client and the server during a transaction over the connection

to facilitate the transaction in the event of the dynamic connection failing whereby avoiding repeating the entire transaction.

12. (Original) The method of claim 6 wherein the connection is a wireless connection.

13. (Currently Amended) A device for preventing blocking of at least one application communicating with a network over a connection, the device comprising[[:]] the at least one application, ~~having~~ the at least one application comprising:

at least one software module for presenting a user interface; ~~and~~

at least one client module for asynchronously communicating with a server;

a media-sense module for detecting whether the connection is operational, the media-sense module configured to, at least:

detect cessation of traffic on a link underlying the connection; and

determine an error rate for the connection;

a first software module for saving a state of the at least one client module; and

a second software module for retrieving the saved state and ~~continue the~~ continuing a communications session when the connection is restored as detected by the media-sense module.

14. (Original) The device of claim 13 wherein the client module receives user input from more than one user interface.

15. (Original) The device of claim 13 wherein the client module transmits data over the connection in response to a media sense event generated by the media-sense module, the media sense event corresponding to establishment of the connection.

16. (Original) The device of claim 13 wherein the client module aborts data transmission over the connection in response to a media sense event generated by the media-sense module, the media sense event corresponding to failure of the connection.

17. (Original) The device of claim 16 wherein the client module stores an interrupted data transmission for subsequent attempts.

18. (Original) The device of claim 16 wherein the client module updates a state of the server, the state corresponding to data to be transmitted over the connection.

19. (Original) The device of claim 16 wherein the client module updates a state of the client module, the state corresponding to data to be transmitted over the connection.

20. (Original) The device of claim 16 wherein the client module updates a state of the server, the state corresponding to data already transmitted over the connection.

21. (Original) The device of claim 16 wherein the client module updates a state of the client module, the state corresponding to data already transmitted over the connection.